$\boxed{\textit{solution}}$

THEORY OF COMPUTATION
CSCI 320, course # 63570
Test # 2
May 6, 2015
instructor: Bojana Obrenić

NOTE: It is the policy of the Computer Science Department to issue a failing grade in the course to any student who either gives or receives help on any test.

Your ability and readiness to follow the test protocol described below is a component of the technical proficiency evaluated by this test. If you violate the test protocol you will thereby indicate that you are not qualified to pass the test.

this is a **closed-book** test, to which it is **forbidden** to bring anything that functions as: paper, calculator, hand-held organizer, computer, telephone, camera, voice or video transmitter, recorder or player, or any device other than pencils (pens), erasers and clocks;

answers should be written only in the space marked "**Answer:** " that follows the statement of the problem (unless stated otherwise);

scratch should never be written in the answer space, but may be written in the enclosed scratch pad, the content of which *will not be graded;*

any problem to which you give **two or more (different) answers** receives the **grade of zero** automatically;

student name has to be written **clearly** on **each page** of the problem set and on the first page of **scratch pad** the during the **first five minutes of the test**—there is a penalty of **at least 1 point** for each missing name;

when requested, **hand in** the problem set together with the scratch pad;

once you leave the classroom, you cannot come back to the test;

your **handwriting** must be legible, so as to leave no ambiguity whatsoever as to what exactly you have written.

You may work on as many (or as few) problems as you wish.
time: 75 minutes.

each **fully** solved problem: 20 points.
full credit: 100 points.

Good luck.

problem:	01	02	03	04	05	06	07	total: [%]
grade:								

Problem 1 Let L be the set of exactly those strings over the alphabet $\{a, b, c, d\}$ that satisfy all of the following properties.

1. the length of the string is equal to $3n + 1$, for some natural number $n \geq 0$;
2. all of the first (leftmost) n symbols are b's;
3. all of the last (rightmost) $2n$ symbols are elements of $\{a, d\}$;
4. the $(n + 1)$st symbol (from the left) is c.

(a) List four distinct strings that belong to L. If this is impossible, state it and explain why.

Advice for Answer: The general template for strings of this language is:

$$b^n\, c\, (a|d)^{2n}$$

(b) Write a complete formal definition of a context-free grammar that generates the language L. If such a grammar does not exist, state it and prove your answer.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\to bSAA \mid c \\ A &\to a \mid d \end{aligned}$$

$\boxed{\textit{solution}}$

(c) Write a complete formal definition of a *regular* context-free grammar that generates the language L. If such a grammar does not exist, state it and prove your answer.

Answer: This grammar does not exist. If L had a regular grammar, then L would be a regular language. However, L is not regular. To prove this, we show that Pumping Lemma does not hold for L.

Observe that all words of L satisfy the following characteristic property: number of b's is equal to one half of the total number of a'and d's.

Assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L. Let $m > k$; then the word $w = b^m ca^{2m}$ belongs to L.

In any "pumping" decomposition such that $b^m ca^{2m} = uvx$, we have: $|uv| \leq k < m$. Hence, the "pumping" substring v consists entirely of b's, say $v = b^\ell$. Recall that $\ell > 0$, since the "pumping" substring cannot be empty. Pump up once, obtaining the word:

$$w_1 = b^{m+\ell} ca^{2m}$$

Since $2m \neq 2(m + \ell)$, word w_1 violates the stated characteristic property and thus $w_1 \notin L$, in violation of the Pumping Lemma.

Problem 2 Let L be the set of exactly those strings over the alphabet $\{a, b, c, d\}$ that satisfy all of the following properties.

1. the length of the string is equal to $3n+1$, for some natural number $n \geq 0$;
2. all of the first (leftmost) n symbols are b's;
3. all of the next n symbols (at positions $(n+1)$ through $2n$ from the left) are c's;
4. all of the last (rightmost) n symbols are d's;
5. the symbol at position $2n+1$ (from the left) is a.

(a) List four distinct strings that belong to L. If this is impossible, state it and explain why.

Advice for Answer: The general template for strings of this language is:

$$b^n\, c^n\, a\, d^n$$

(b) Write a complete formal definition of a context-free grammar that generates the language L. If such a grammar does not exist, state it and prove your answer.

Answer: Such a grammar does not exist, since L is not context-free. To prove this, we show that Pumping Lemma does not hold for L.

Observe that every word of L satisfies the following characteristic property: the number of b's is equal to the number of c's and equal to the number of d's, and there is exactly one a.

Assume the opposite, that L is context-free. Let k be the constant as in the Pumping Lemma for L. Select $n > k$ and consider a word $w = b^n\, c^n\, a\, d^n \in L$, which must pump. If the pumping window is positioned so that a's are pumped, the new word will have more than one a, violating the characteristic property. Otherwise, in any pumping decomposition, the pumping window is entirely within one of the following three segments: b^n, c^n, d^n or it spans two adjacent segments, since the pumping window is too short to extend into more than two of them, because its length is less than k, and thereby less than n. Thus, if we pump up once, there will be at least one of the segments b^n, c^n, d^n where pumping occurs and at least one where it does not occur. The number of occurrences of at least one of the letters b, c, d will have increased, while the number of occurrences of at least one other of these letters will not have changed. The new word will violate the characteristic property and thus will not belong to L. Since L violates the Pumping Lemma, L is not context free.

$\boxed{\textit{solution}}$

(c) Draw a state-transition graph of a finite automaton M that accepts the language L. If such an automaton does not exist, state it and prove your answer.

Answer: This automaton does not exist, since L is not regular. If L was regular, then L would be content free, because every regular language is content free. However, we prove in part (b) that L is not context free, hence L cannot be regular.

Problem 3 Let L be the set of exactly those strings over the alphabet $\{a, b, c, d\}$ that satisfy all of the following properties.

1. the length of the string is equal to $3n + 1$, for some natural number $n \geq 0$;
2. all of the first (leftmost) $3n$ symbols are equal to each other;
3. none of the first (leftmost) $3n$ symbols is d;
4. the last symbol (at position $3n + 1$ from the left) is d.

(a) List four distinct strings that belong to L. If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (c).

(b) Write a complete formal definition of a context-free grammar that generates the language L. If such a grammar does not exist, state it and prove your answer.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, K\}$, and the production set P is:

$$
\begin{aligned}
&S \to Ad \mid Bd \mid Kd \\
&A \to \lambda \mid AA \mid aaa \\
&B \to \lambda \mid BB \mid bbb \\
&K \to \lambda \mid KK \mid ccc
\end{aligned}
$$

$\boxed{\textit{solution}}$

(c) Write a regular expression that defines L. If such a regular expression does not exist, state it and prove your answer.

Answer:

$$((\boldsymbol{aaa})^* \cup (\boldsymbol{bbb})^* \cup (\boldsymbol{ccc})^*)\ \boldsymbol{d}$$

Problem 4 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c, A\}$; $Q = \{q, p, t, v, x, y, s, z, m, e\}$; $F = \{m\}$; and δ consists of the following transition set:

$[q, a, p, A, R]$	$[t, a, t, a, R]$	$[s, a, s, a, L]$
$[q, b, e, A, R]$	$[t, b, t, b, R]$	$[s, b, z, b, L]$
$[q, c, p, A, R]$	$[t, c, t, c, R]$	$[s, c, s, c, L]$
$[q, B, e, B, R]$	$[t, B, v, B, L]$	
		$[z, a, m, a, L]$
$[p, a, t, A, R]$	$[v, a, x, a, L]$	$[z, b, z, b, L]$
$[p, b, e, A, R]$	$[x, c, y, c, L]$	$[z, c, z, c, L]$
$[p, c, t, A, R]$	$[y, b, s, b, L]$	
$[p, B, e, B, R]$		$[e, B, e, B, R]$

Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) M accepts by final state. B is the designated blank symbol.

Let L be the set of strings that the Turing machine M *accepts.* Let L^∞ be the set of strings on which the Turing machine M *diverges.*

(a) List four distinct strings that belong to L. If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (b).

(b) Write a regular expression that defines L. If such a regular expression does not exist, prove it.

Answer:

$$(\boldsymbol{a} \cup \boldsymbol{c})(\boldsymbol{a} \cup \boldsymbol{c})(\boldsymbol{a} \cup \boldsymbol{b} \cup \boldsymbol{c})^* \, \boldsymbol{a} \, (\boldsymbol{b} \cup \boldsymbol{c})^* \, \boldsymbol{b} \, (\boldsymbol{a} \cup \boldsymbol{c})^* \, \boldsymbol{bca}$$

$\boxed{\textit{solution}}$

(c) List four distinct strings that belong to L^∞. If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (d).

(d) Write a regular expression that defines L^∞. If such a regular expression does not exist, prove it.

Answer:

$$\boldsymbol{\lambda} \cup \boldsymbol{a} \cup \boldsymbol{b} \cup \boldsymbol{c} \cup \boldsymbol{ab} \cup \boldsymbol{cb}$$

(e) Is L recursively enumerable? Explain your answer.

Answer: Yes, by definition, since L is accepted by Turing Machine M.

Problem 5 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q)$ such that: $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1, N\}$; $Q = \{q, p, t, v, x, y, s, m\}$; and δ consists of the following transition set:

$$\begin{array}{lll}
[q,0,p,Z,R] & [t,0,t,0,R] & [s,0,s,0,R] \\
[q,1,p,N,R] & [t,1,t,1,R] & [s,1,s,1,R] \\
[q,B,m,B,R] & [t,B,v,B,L] & [s,B,s,B,R] \\
\\
[p,0,t,0,R] & [x,0,x,0,L] & [y,0,y,0,L] \\
[p,1,t,1,R] & [x,1,x,1,L] & [y,1,y,1,L] \\
[p,B,m,B,R] & [x,Z,s,0,R] & [y,N,s,1,R] \\
\\
[v,0,x,0,L] & & \\
[v,1,y,1,L] & &
\end{array}$$

Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.

Let L be the set of strings on which the Turing machine M *halts*. Let L^∞ be the set of strings on which the Turing machine M *diverges.*

(a) List four distinct strings that belong to L^∞. If this is impossible, state it and explain why.

Advice for Answer: L^∞ is defined by the following regular expression.

$$\mathbf{0(0 \cup 1)^*0 \ \cup \ 1(0 \cup 1)^*1}$$

(b) Draw a state-transition graph of a finite automaton M that accepts the language L^∞. If such an automaton does not exist, state it and prove your answer.

Answer: See Figure 1.

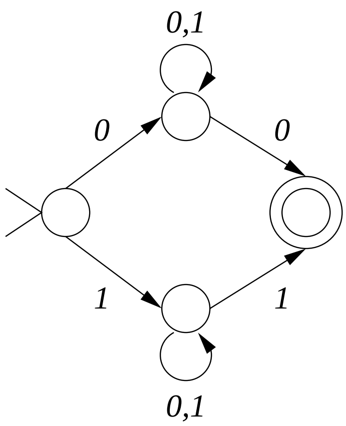

Figure 1:

$\boxed{\textit{solution}}$

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A string $w \in \Sigma^*$.

OUTPUT: **yes** is $w \in L^\infty$; **no** otherwise.

If this algorithm does not exist, prove it.

Answer: Simulate the finite automaton given in the answer to part (b) and accept exactly when it accepts.

(d) Is L^∞ decidable? Explain your answer.

Answer: Yes—the algorithm given in the answer to part (c) decides L.

Problem 6 Let L_1 be the language accepted by the pushdown automaton $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where $\Sigma = \{a, b, c, d, e\}$; $\Gamma = \{A, B\}$; $Q = \{q, t, s\}$; $F = \{s\}$; and the transition function δ is defined as follows:

$$[q, b, \lambda, q, BBB]$$
$$[q, \lambda, \lambda, t, \lambda]$$
$$[t, d, \lambda, t, AA]$$
$$[t, e, \lambda, s, \lambda]$$
$$[s, c, A, s, \lambda]$$
$$[s, a, B, s, \lambda]$$

(M is defined so as to accept by final state and empty stack.)

(a) List four distinct strings that belong to L_1. If this is impossible, state it and explain why.

Advice for Answer: The general template for strings of this language is:

$$b^n \, d^k \, e \, c^{2k} \, a^{3n}$$

(b) Write a complete formal definition of a context-free grammar that generates L_1. If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S, A\}$, and the production set P is:

$$S \to bSaaa \mid A$$
$$A \to dAcc \mid e$$

$\boxed{solution}$

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A deterministic Turing Machine M which accepts a language $L(M)$.

OUTPUT: **yes** if $L(M) \cap L_1 = \emptyset$; **no** otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of Turing machine whose languages satisfy the property:

has an empty intersection with L_1.

This property is nontrivial, since it is true say for empty set (because $\emptyset \cap L_1 = \emptyset$), but false for say Σ^* (because $\Sigma^* \cap L_1 = L_1 \neq \emptyset$.) By Rice's Theorem a Turing machine cannot exist that decides the set of Turing machines whose languages satisfy any nontrivial property.

(d) State a nontrivial property of recursively enumerable languages that is true for L_1 but false for $\boldsymbol{a}^*$. Explain why the property is nontrivial and show that it indeed is true for L_1 but false for $\boldsymbol{a}^*$. If such a property does not exist, state it and explain your answer.

Answer: "Does not contain the empty string." This property is true for L_1, since every string in L_1 contains one e, which is proved by inspection of M. This property is false for $\boldsymbol{a}^*$, because $\boldsymbol{a}^*$ contains the empty string, by definition of Kleene star. Since it is true for one recursively enumerable language (L_1) but false for another ($\boldsymbol{a}^*$), the property is by definition non-trivial.

Problem 7 Let L_1 be the language accepted by the pushdown automaton $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where $\Sigma = \{a, b, c, d, e\}$; $\Gamma = \{A, B\}$; $Q = \{q, t, v, x, s\}$; $F = \{s\}$; and the transition function δ is defined as follows:

$$\begin{array}{l} [q, b, \lambda, q, BB] \\ [v, d, \lambda, v, AAA] \\ [t, c, B, t, \lambda] \\ [x, a, A, x, \lambda] \\ [q, \lambda, \lambda, t, \lambda] \\ [t, \lambda, \lambda, v, \lambda] \\ [v, \lambda, \lambda, x, \lambda] \\ [x, e, \lambda, s, \lambda] \end{array}$$

(M is defined so as to accept by final state and empty stack.)

(a) List four distinct strings that belong to L_1. If this is impossible, state it and explain why.

Advice for Answer: The general template for strings of this language is:

$$b^n\, c^{2n}\, d^k\, a^{3k}\, e$$

(b) Write a complete formal definition of a context-free grammar that generates L_1. If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{array}{l} S \to ABe \\ A \to bAcc \mid \lambda \\ B \to dBaaa \mid \lambda \end{array}$$

solution

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A deterministic finite automaton M which accepts a language $L(M)$, and a string $w \in \Sigma^*$.

OUTPUT: **yes** if $w \in L(M) \cap L_1$; **no** otherwise.

If this algorithm does not exist, prove it.

Answer: Recall that the intersection of a context free language with a regular language is context free. Apply the known algorithm to construct a pushdown automaton M_X that accepts the intersection of the context free language L_1 (accepted by the original pushdown automaton) and the regular language $L(M)$ (accepted by the argument finite automaton M.) Simulate M_X on w and return exactly what M_X returns.

(d) State a trivial property of recursively enumerable languages that is true for L_1 but false for $(\boldsymbol{aa})^*$. Explain why the property is trivial and show that it indeed is true for L_1 but false for $(\boldsymbol{aa})^*$. If such a property does not exist, state it and explain your answer.

Answer: Such a property does not exist. If a property of recursively enumerable languages was true for L_1 but false for $(\boldsymbol{aa})^*$ then such a property would not be trivial, because by definition a trivial property has the same value for all recursively enumerable languages.